Argonne NATIONAL LABORATORY

Magellan for HPC Users

Narayan Desai

desai@mcs.anl.gov

Mathematics and Computer Science Division

Argonne National Laboratory

U.S. DEPARTMENT OF ENERGY

Talk Overview

- Project Goals
- System Architecture
- Software Architecture
  - Virtualization
  - Application Runtime Environment
  - Resource Management
  - I/O
- Basic System Use Differences

The Magellan Project

- Determining suitability of clouds for HPC
- Addressing infrastructure and system software shortfalls
- Getting experience running cloud resources
- Adapting applications to cloud programming models

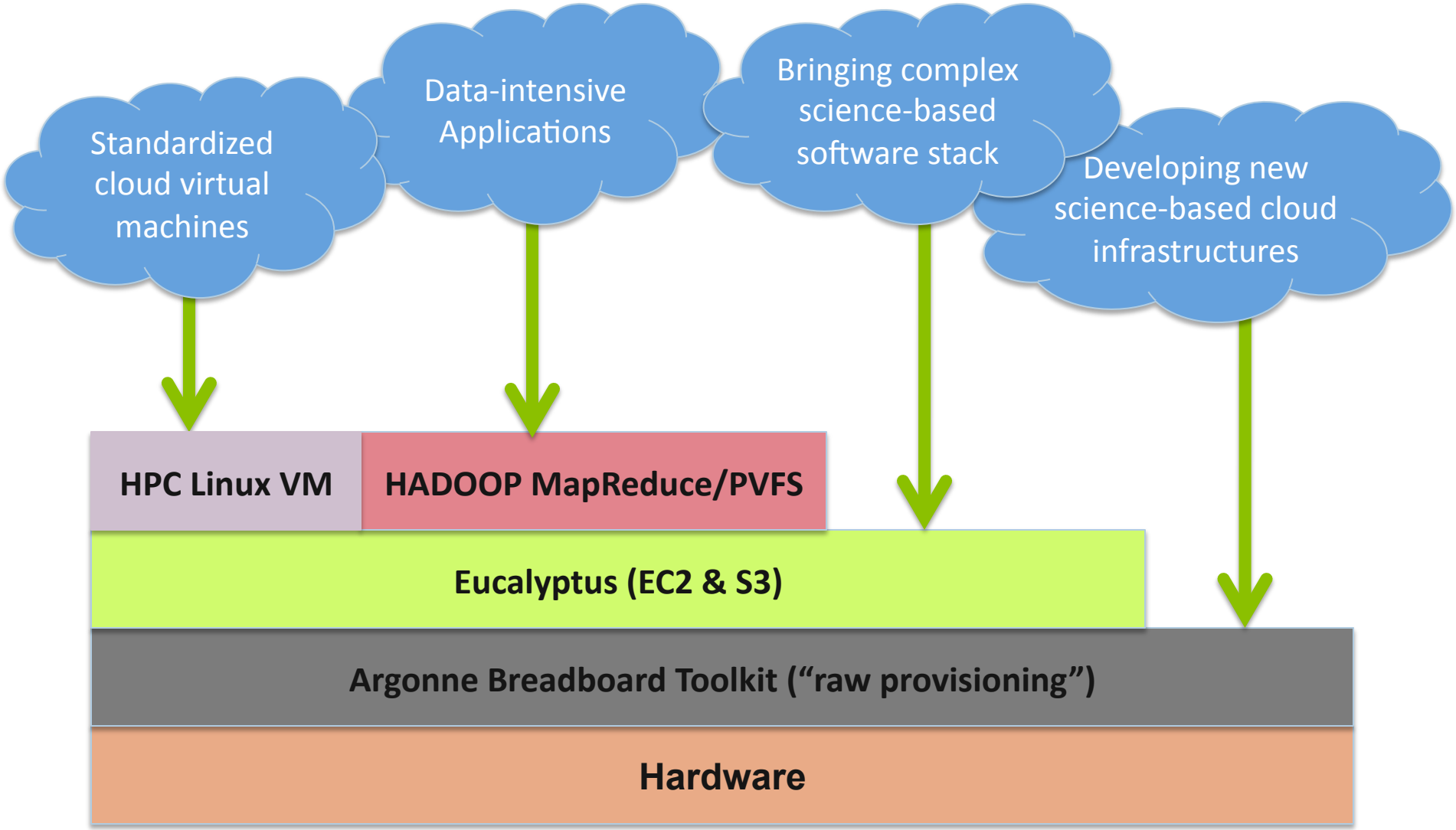
Traditional HPC infrastructure use case

- Operated as a coherent, persistent infrastructure
  - Includes computing resources
  - Storage
  - Login resources
  - Archival Storage
  - System support infrastructure (administration, application engineers)
- Centrally controlled environment
  - Users adapt to the provided software environment
  - Users share a single environment
- Users login to a (usually) Linux compilation/job submission node
- Users submit work to a batch queue
- Resource management is sophisticated, balancing system utilization, per job response times, fairshare, and other system metrics
- Jobs have a finite (and limited) lifespan

Magellan is quite different

- Designed to support experimentation
- Users control their software environment
  - Root access
  - User supplied kernel and OS
- Limited persistent infrastructure
  - 2 Management Nodes
  - 4 Cloud Infrastructure Nodes
  - 8 Storage Servers
- The balance of the system is user provisionable
- Maximal flexibility, at the cost of integration and support
  - Systems aren't auto-configured into a coherent infrastructure automatically
- Virtualization engrained in the system software architecture
  - At least for basic users

Magellan System Software Architecture



Magellan User Model

- Virtualization via EC2 interfaces primary interface
  - For now, raw provisioning to follow
- No persistent login infrastructure
- All interactions occur with the Eucalyptus via the network.
  - VM Instance requests
  - S3 Storage access
- Login resources can be carved out of allocations, if needed
- Instances charged against allocation based on configuration
  - Several configurations available
    - Range from 1 core/1GB to full nodes
  - Larger configurations cost more

Magellan Resource Management

- Rudimentary compared with traditional batch schedulers
  - No Queue (!)
  - All requests satisfied at once, or not at all
  - No priorities, fairshare
  - No reservations
- Utilization maximized at the cost of everything else
  - Designed to maximize resource occupancy
  - (by paying customers)
- Avoids a lot of the politics of traditional HPC scheduling
  - Everyone pays the same price, so you don't

Security

- On a system like this, security needs to mediate in three ways
  - Protecting users from one another
  - Protecting the system from the users
  - Protecting the world from the users
- Users are automatically sandboxed
  - Different IP spaces
  - Firewalling rules
- Resources are reclaimed and reused, so policies based on IP addresses are a bad idea
- EC2 (and Eucalyptus) have interfaces to control IP firewalling
  - They need to be used carefully
  - Open ports are exposed to the internet
- The bottom line
  - Be careful, think about what you're doing, and ask questions

Support

- Because of the variability in configurations, traditional system support is not feasible
- Leads to a DIY sort of ethic
- Much like Unix software compilation in the old days
  - One user figures things out, and others can use reuse their OS images, software stacks

Virtualization

- Magellan uses KVM
- Each node instance gets reserved resources
  - Rick will provide details as to instance configurations
- Infiniband will be available to guests soon
- All ethernet networking is bridged
- Takes 3-5% of raw node performance
- Support for exotic hardware forthcoming
- As Susan mentioned, impact on OS jitter is still unknown
- I/O performance suffers due to additional buffer copies

Software Stability

- The HPC system software stack is largely robust
  - Production tested
  - Occasional issues, not pervasive
- The whole cloud software stack is immature
  - We find major issues with Eucalyptus about once a week
  - (like yesterday)
- Much of this software is under heavy development
- Rapidly changing
  - Feature sets
  - Hopefully bugfixes
- Translates to occasional outages

Experimentation Architecture

- Public Eucalyptus instance
  - Includes the majority of compute hardware
  - Running a stable version of system software
- Development Eucalyptus instance
  - For testing and system development
  - Hypervisor improvements
  - System software testing
  - Moderate number of nodes
- User cloud instances
  - Used to do cloud software development or modification
  - Reserved persistent infrastructure node per instance
  - Small numbers of nodes, dynamically allocated

Questions?